

UNITED STATES UTILITY PATENT APPLICATION

FOR

**REMOTE AUTOMATED VOLUME MOUNTING**

Inventor:

**Scott Ryder**

Prepared by:

Anand Sethuraman  
STATTLER, JOHANSEN & ADELI LLP  
P.O. BOX 51860  
PALO ALTO, CALIFORNIA 94303-0728  
FAX: (650) 934-0475

## REMOTE AUTOMATED VOLUME MOUNTING

Inventor: **Scott Ryder**

### BACKGROUND

#### 5        1.    Field of the Invention

The present invention relates to software for computer systems and networking. More specifically, the present invention relates to the accessibility of remote storage devices.

#### 2.    Description of Related Art

10        Disk drives, logical or physical partitions of disk drives and even file servers are all often referred to loosely as "volumes". Computer operating systems typically have commands and procedures for "mounting" disk drives (such as floppy drives, hard drives and CD (Compact Disc) drives) to make them available  
15        to the user. On a local machine such mounting can be done once during the installation of the drive and thereafter will be accessible upon any system startup instance. Drives may also be "hot-swapped" so that they are recognized by the operating system and made available to the user instantly upon interfacing with  
20        the computer's hardware.

Over a local area network (LAN), client machines can connect to volumes hosted by servers over the LAN depending on their authentication and permissions. To connect or mount such networked volumes, one common requirement is that the client be  
25        authenticated by the server. Once so authenticated, the user must search or browse for hosted volumes, and then connect to the desired volumes often using yet another authentication procedure.

For instance, on Appleshare(TM), a product of Apple Computers, Inc., the client user browses the network and finds first the desired server. Then, the user clicks on the server icon and is usually required (where guest access is not enabled on the  
5 server) to again enter a username and password. Once authenticated, the user must browse and select from available volume(s). Thereafter, the selected volume(s) is(are) available on the operating system desktop of the client machine appearing as any local drive would.

10 One issue with typical volume mounting is the time expended. Another is that of repeating the same process the next time the same volume needs to be mounted. Any of the parameters, such as a network location, name or alias, that uniquely identify a given volume may have changed or may not be remembered by the user and  
15 must be found again. Also, there are often additional steps needed for clients that remotely "dial-in" to a given network, and then attempt to locate network resources.

There is thus a need for automating the volume mounting process.

**SUMMARY OF THE INVENTION**

The invention is a method and apparatus for remotely and automatically mounting volumes over a network. The invention makes use of an interface, such as a web browser, to connect a client to a server. Volume mounting parameters are retrieved from generic or specific profiles that are accessed in accordance with a request from the client to the server. The server supplies volume mounting parameters specific to the client via a plug-in installed on the client. The plug-in then interacts, in one or more embodiments of the invention, with an application programming interface and/or system software on the client which automatically mounts the volume(s) in accordance with the supplied parameters.

In at least one embodiment of the invention, the server assembles a mounting command block consisting of retrieved volume mounting parameters and a mount command. Additionally, in some embodiments, the plug-in must first verify as approved the server returning the mounting command block in order for volume mounting to proceed.

**BRIEF DESCRIPTION OF THE DRAWINGS**

**Figure 1** illustrates remote automated volume mounting according to at least one embodiment of the invention.

**Figure 2** is a flowchart of retrieving a user mounting  
5 profile according to at least one embodiment of the invention.

**Figure 3** illustrates a sample set of user mounting profiles that may be retrieved from a server according to the invention.

**Figure 4** is a flowchart of client-server interaction according to at least one embodiment of the invention.

10 **Figure 5** illustrates the operation of the plug-in in accordance with one or more embodiments of the invention.

**Figure 6** shows an exemplary computer system acting as a client in accordance with the present invention.

15 **Figure 7** shows an exemplary computer system when acting as a server in accordance with the present invention.

**DETAILED DESCRIPTION OF THE INVENTION**

**Figure 1** illustrates remote automated volume mounting according to at least one embodiment of the invention. Volume mounting is commenced, according to the invention, by a client logging into a control web server and, optionally, sending authentication information (such as a login identification and password) to the web server. **Figure 1** shows two such clients, client 100 and client 104 capable of communicating with a web server 150. Each client 100 and 104 has plug-in modules 101 and 105, respectively, installed on their systems which communicate with web server 150. In one embodiment, to maintain security, the optional authentication information is accepted by the web server 150 only when a cookie is detected on the client. The existence of the cookie indicates that the appropriate plug-in module has been installed on the client.

A user of client 100 who wants to have volumes mounted from a remote location may send authentication information to web server 150. Alternatively, some process or event occurring on client 100 may require/request to have one or more remote volumes mounted as part of its operation. Web server 150 maintains or has access to a profile of client 100 regarding what file volumes should be automatically mounted, which volumes, if any, should be selectively mounted and the client's authentication information. If there is authentication information from the client 100, web server 150 compares the authentication information provided by client 100 with the profile maintained for the client 100. If the authentication is approved, web server 150 sends the plug-in

101 of client 100 mounting command & parameters such as which  
volumes to mount to, and username and password information  
specific to those volumes. For instance, in the case of client  
100, web server 150 would send information in order to mount a  
5 volume 160, such as the file server alias, volume name(s),  
location/designation and access codes (such as logins and  
passwords), if any, that would be required.

Plug-in 101 performs a number of functions prior to the  
automatic mount specified in the mounting command & parameters.

10 The plug-in 101 obtains the address (such as IP (Internet  
Protocol) address) of the web server 150. Then, the plug-in 101  
checks to see if the address of web server 150 is included in a  
list of approved servers. The list of approved servers is  
embedded in the plug-in 101 or encrypted/scrambled such that only  
15 plug-in 101 may access. If the web server 150's address is not  
valid (i.e. does not appear in the list of approved servers), the  
plug-in 101 terminates its operation and no mounting occurs.  
This serves as an additional level of security in addition to the  
optional user authentication. With the approved server address  
20 mechanism, the plug-in 101 can be better assured that the  
potential volume mount is authorized, with or without user  
authentication.

If the web server 150's address is valid, then file system  
software built on the operating system of client 100 receives the  
25 mounting command & parameters and automatically mounts the  
specified volume (volume 160) without the need for user  
intervention. After volume 160 is mounted successfully, it will

thereafter behave as if it were a native file volume (e.g. hard disk connected directly to client bus) of client 100. For instance, the icon pertaining to volume 160 may be displayed of the desktop or other location where the list of all other disks and volumes of client 100 are displayed.

Like client 100, client 104 is also equipped with an appropriate plug-in module 105. When client 104 accesses the website of server 150, then server 150 may accept optional user authentication information from client 104. If the profile corresponding to the login ID supplied by client 104 indicates a need to connect automatically with volumes 160 and 162, the mount command & parameters for both of these volumes is sent in turn to the operating system of client 104 via plug-in 105. The client 104 then proceeds to mount both volumes 160 and 162 without any further intervention by users of client 104. Plug-in 105 also checks to see if the web server 150 is on its list of approved servers prior to accepting the mount command & parameters from web server 150. The list of approved servers in each of the plug-ins 101 and 105 may be different, but are shown in Figure 1 to both include, at least, the address of web server 150.

As demonstrated in **Figure 1**, the invention provides for a single easily accessible point of entry, e.g. web server 150, to mount volumes that are to be shared among clients. Volumes 160 and 162 appear to client 104 as would native volumes of client 104.

Though not specifically depicted in **Figure 1** it can be assumed that web server 150, client 100, client 104, volume 150



and volume 160 could be connected together through some global network such as the Internet, WAN (wide area network) or LAN (local area network). In accordance with the invention, the network configuration is not a consideration in mounting of the file volume. However, if an IP (Internet Protocol) address is required to connect to a given volume, then that address can be obtained by resolving a supplied server alias to an IP address. IP address, host name and other designation of network location (such as local area network routing path) may be transmitted along with volume mounting parameters or may be resolved by translating aliases in a table. A successful mounting of the volume(s) implies that the client is enabled with the appropriate transport/network mechanisms necessary for direct transfer of data with those volumes.

Further, while not illustrated in **Figure 1**, the web server 150 may be connected to a number of other servers or have processes/applications running upon it that perform functions such as database management.

**Figure 2** is a flowchart of retrieving a user mounting profile according to at least one embodiment of the invention. First, the user of the client machine should point the browser or other web interface application to the web server website (block 210). The web server website address can be provided during plug-in installation via an automatic addition to the browser's bookmark listing (such as the 'Favorites' list found on some browsers), or by placing a shortcut/menu item on the client's desktop. When the server website is contacted to, it checks for

the existence of a particular cookie on the client machine (block 220). A cookie is a persistent information store (file/object) stored on a client that can be accessed by websites the client connects to. The cookie being checked for exists only if the  
5 plug-in has been installed or is part of the system.

If the cookie does not exist, then the session is terminated (block 280) in order to preserve security. If the cookie exists (i.e. the plug-in is installed in the client), then the login process is commenced. The server website then requests a login

10 ID and password or similar authentication from the client (block 230). In response to the client sending the server

authentication information, the server checks if the authentication is valid (block 240). If the authentication is not valid then the session is terminated (block 280). If the

15 authentication is valid, then a profile associated with the authenticated login ID is retrieved (block 250). The profile indicates what volume(s) are to be mounted, what usernames and passwords (if any) are to be supplied for those volumes, and the network location or designation of the volumes (such as server

20 alias, host name, IP address, routing path, etc.). Additionally, the profile may indicate folders within volumes where access is restricted or which folders are public/private. The retrieved profile may then be parsed, and its information supplied to the plug-in of the client (as part of the volume mounting command  
25 block, for example).

**Figure 3** illustrates a sample set of user mounting profiles that may be retrieved from a server according to the invention.

A profile datastore is a set of profiles stored on the server that is used to regulate the remote automated volume mounting process. The profile datastore 300 of **Figure 3** depicts profiles for two separate login IDs—"john" and "jack". Profile datastore 300 may be a part of a database, portion thereof or a flat file stored on or made accessible to the volume mounting website server. Profile datastore 300 consists of variable length records, each record characterized primarily by a login ID field 313. The size/length of each record depends directly upon the number of volumes to be mounted for that login ID.

The authentication information (see **Figures 1** and **2**, and associated description above) which may be sent to the server by the client is compared against the authentication strings in the profile datastore 300. The authentication strings include login ID 313 and a password field 315. Where user authentication is transmitted, to gain access to the website server and volume mounting parameters, both the login ID and password must match the corresponding authentication strings. For example, the password "yyyyyy" must be transmitted along with the login ID "john" if login ID "john" is to be authenticated.

Assuming the server authenticates login ID "john", then volume mounting parameters contained the record indexed by login ID "john" is retrieved. The volume mounting parameters includes a number of fields, including, volume name 330, server alias 340, username 350 and password 360. The volume name field 330 is a designator used to easily identify the volume and may be descriptive in nature. Client operating environments may use the

value of the volume name field 330 as the name of the volume once mounted. The value of the volume name field 330 can be used by the client in constructing a desktop icon for the mounted volume. The value of the server alias 340 is a string, which may be

5 descriptive in nature, designating the server where the volume name specified in the corresponding volume name field 330 is physically a part of. A table can be provided which resolves server alias 340 to an actual Internet Protocol (IP) address where the server is located. For instance, the server alias  
10 "doghouse" may be resolved by a look-up table to the physical IP address "128.127.1.1". Alternatively, depending on the network configuration through which volume mounting is being performed, the value of the server alias field 340 may be resolved/translated to a local area network designation or  
15 network routing path. For instance, where only intranet-based clients are permitted to mount volumes, then IP addresses may not be needed, or may not even be used.

The values of the volume name 330 and server alias 340 fields are passed to a client via the client's plug-in. The  
20 plug-in also receives values for a username 350 and password 360 associated with a particular volume. In the record indexed by login ID "john", that particular ID "john" has a username 350 of "jgray" and a password 360 of "xxxxxx". The plug-in takes the username "jgray" and associated password "xxxxxx" and  
25 automatically uses these as arguments when the command is issued to mount volume A. This eliminates the need for the user to manually enter/remember "jgray" and "xxxxxx" or the need to

manually clear (by clicking, hitting a key) a dialog box that may store the correct username and password ("jgray" and "xxxxxx", respectively) for volume A. The volume mounting parameters for each volume includes a username field 350 and password field 360 separate from that of the login ID 313 and password 315 since these may differ. Similarly, volume name 330, server alias 340, username 350 and password 360 are all transmitted to the client plug-in after being retrieved from profile datastore 300. The volume mounting parameters for volume C may be transmitted following the mounting/attempted mounting of volume A, or may be transferred to the client and queued thereon. Depending upon the client's operating environment, it may be possible to mount both of the volumes A and C in a concurrent fashion.

With the use of profile datastore 300, the volume mounting preferences (information) pertaining to multiple users may all be centralized in one location. For instance, a second user, having a website login ID of "jack" can also mount volume A automatically upon entering in the proper password "111111" associated therewith. In this example, the username 350 and password 360 ("jack" and "111111" respectively) are the same as the website login ID 313 and password 315 (again, "jack" and "111111" respectively). In other embodiments, a given login ID may also be able to manually select a list of volumes that can be mounted.

Such a profile datastore 300 may also store, in addition to specific user profiles, one or more generic profiles that correspond to certain client processes or events that require a

mounting of volumes. Such generic profiles may be retrieved from datastore 300 without a validated user authentication. Generic profiles may be useful where public or global access to certain volumes needs to be granted.

5           **Figure 4** is a flowchart of client-server interaction according to at least one embodiment of the invention. A client can request a web server for a volume mount either as the result of a successful user authentication (block 410) or a situation, event or process occurring on the client that requires a volume  
10 mount (block 420). For instance, where a process such as file uploading requires that a certain volume be available to transfer data, the request for mount could be automatically placed, without user authentication, to the server. The mount request is not necessarily explicit but a logical means of describing the  
15 initiation of the automated mounting process. As a result of either blocks 410 or 420, profiles for the user/event are retrieved by the server (block 430). As mentioned above, the profiles may reside in one or more datastores accessible by the server, and contain volume mounting information indexed by user.  
20 Additionally, a subset of such profiles may also be marked or indexed as generic and associated with a process/event rather a particular user. The profile(s) that are retrieved as a result of block 430 supply mounting preferences 435 for the process to continue. Mounting preferences 435 include items such as volume  
25 name(s), username(s), password(s) and server alias(es). Using the preferences 435 as input, the server assembles a "mounting command block" (block 440) for the plug-in on the client.

The mounting command block includes volume mounting parameters parsed from the preferences 435 and a mounting command set. A mounting command block may be assembled as part of an XML (Extensible Markup Language) or similar document and may be represented in the form of tags. The mounting command block, either separately, or as part of an XML document, is then sent to the client (block 450). The client receives the command block via the plug-in that was previously installed or embedded thereon.

**Figure 5** illustrates the operation of the plug-in in accordance with one or more embodiments of the invention. After the mounting command block is issued by a server, the plug-in on the client obtains the address of that server (block 510). The mounting command block, which consists of mounting profile and the mount command, is passed as an input to the plug-in (block 515). The plug-in accesses a list of servers approved to issue such mounting commands that may be embedded within it (block 517). The plug-in checks whether the address of the server that issued the mounting command block is on the built-in list of approved servers (block 520). If it is not, then the process terminates and no further action is taken by the plug-in.

If the server that issued the mounting command is on the built-in approved server list, then the mounting process can continue. The plug-in next parses the mounting profile of the mounting command block for the volume mounting parameters, including the server alias (of where the volume resides), the volume name, and any associated username and password (block

530). The plug-in then utilizes a mapping table (block 535) that translates/resolves server aliases into real, physical addresses such as IP addresses. Using this mapping table, the plug-in translates the parsed server alias to the physical server address (block 540). Through available APIs (Application Programming Interfaces), the plug-in then initiates the mount of the volume specified in the mounting command block that has been parsed (block 550). Examples of such APIs include AFP (Apple Filing Protocol), which works over IP, and CIFS/SMB (Common Internet File System/System Messaging Block) which is found in Microsoft Windows-based machines.

**Figure 6** shows an exemplary computer system acting as a client in accordance with the present invention. A client computer system 600 may be a mobile computer, desktop computer or any general or special purpose information processing device. The client system 600 features a system bus 613 for allowing components, such as a processor 612 and memory 611, to communicate with each other. Memory 611 consists of random access memory (RAM) or other volatile storage mechanism and operates to store instructions to be executed by processor 612 and data. Memory 611 also may be used for storing temporary variables or other intermediate result data during execution of instructions by processor 612.

Client system 600 also has a bridge 614 which couples to an I/O (Input/Output) bus 615. I/O bus 615 connects to system 600 various peripheral and I/O devices such as a network interface card (NIC) 616, a disk 618 and a CD-ROM (Compact Disc-Read Only



Memory) 617. I/O bus 615 also couples to the system a display device 620 such as a monitor or LCD (Liquid Crystal Display). The NIC 616 enables the client 600 to connect to one or more networks 690 such as the Internet or a LAN (Local Area Network) and facilitates the exchange of information between client 600 and other devices/systems on that network 690. Disk 618 is a hard disk drive, floppy disk drive or tape drive or other form of non-volatile storage and is used to store data/application files and other more permanent information.

Client 600 has an operating system software (not depicted) that handles the interaction between its components and other applications. The mounting of disks and the initialization of devices are typical responsibilities of operating system software. Operating system software also governs the manner/mechanism of input/output for users of client 600. Applications, such as a web browser or web interface, are usually built on top of and work in conjunction with the operating system software. Due to its criticality, operating system software is most often permanently stored on disk 618, firmware or other semi-permanent storage. Operating system software may also include extensions, available in APIs, that perform specified tasks such as mounting of a volume over IP (such as Apple Filing Protocol). File system software (such as Apple Computer's Appleshare(tm)) is typically considered a part of the operating system but may also be a separate component software built on top of the operating system software. Such file system software may

also be accessed in addition to or instead of APIs as part of the automated mounting process.

In accordance with one embodiment of the invention, client 600 commences remote automated volume mounting by opening a web browser or web interface to the server website which is accessible over network 690. A connection to the server website is made by establishing an HTTP (Hypertext Transport Protocol) session with the website through the NIC 616. Once established, the website checks for the existence of a cookie on the client indicating that the appropriate plug-in has been installed. Client 600 contains one such cookie 670, stored on disk 618, whose existence signals that the plug-in module has been installed. The plug-in module is dedicated piece of software that can be loaded into memory 612 and passes information to an API/file system software to mount volumes. Once the cookie 670 has been detected by the server, the server requests (via HTTP, secure HTTP or similar mechanism) the client 600 for a login ID and password. In response, a user of client 600 may enter a login ID and password through some input device such as a keyboard (not depicted) or by simply indicating acceptance of a login ID and password stored on client 600. If the login ID and password is authenticated by the server, the client 600 receives therefrom volume mounting command block including volume mounting parameters.

In other embodiments, the server website may be accessed by the operation of some event or situation, instead of strictly as a function of a manual HTTP session. In such embodiments, such

user authentication may be completely bypassed. For instance, where a public file volume is to be downloaded from, it may necessary to mount that volume without any user authentication. In this case, generic mounting parameters could be returned to  
5 the client 600.

A plug-in 685 may be loaded into memory 611 as a part of the operation of web browser software as the result of the event/situation requiring a volume mount. The plug-in 685 consists of one or more portions of computer program code that  
10 load into memory 611 through an application such as a web browser or as the result of some system event. Whether the mounting request is initiated by user authentication or some event or situation, a plug-in 685 first checks to see if the mounting command block was issued by a server that is on the plug-in 685's  
15 built-in list of approved servers. If so, the plug-in 685 parses the volume mounting parameters (see **Figure 5**), resolving server aliases if appropriate, and then passes it to an API or file system software which is either a part of or built on top of the operating system software. In the example of **Figure 6**, a  
20 successful authentication or some other event by the server will cause the plug-in 685 of client 600 to receive mounting parameters regarding networked volume 650. Networked volume 650 is a file system, disk or portion thereof that has been connected to network 690. One parameter pertaining to networked volume 650  
25 is a designation of its volume name as "volume B". The volume name parameter, along with a username and password for the volume corresponding to the authenticated login ID is passed to the

API/file system software through the plug-in 685. For the  
API/file system software of client 600 to mount networked volume  
650, the physical location of the server must be made known. The  
plug-in 685, in one embodiment, will resolve a server alias,  
5 supplied as part of the volume mounting parameters, to a physical  
location such as IP address or network location. In this  
instance, the server alias will be resolved to the IP  
address/network path of networked volume 650.

The file system software connects to networked volume 650  
10 (using the resolved location designation of the server housing  
the volume to be mounted) and then automatically supplies the  
username and password through the file system software. The  
networked volume 650 uses the supplied username and password to  
log client 600 in. Once logged, the operating system software is  
15 signaled and uses the volume name "volume B" and places an icon  
for volume B onto the list of available devices 680. The icon  
and name for networked volume 650 appear on display 620 as part  
of the device list 680. Device list 680 enumerates volumes which  
are either native or appear native through automated mounting  
20 process. Disk 618 which is locally connected through I/O bus 615  
is considered a true native volume, and its volume name "volume  
A" and icon appear in list 680. Likewise, volume B, which is the  
designation of networked volume 650, also appears in the list  
680, indicating that the networked volume 650 can be written  
25 to/read from in the same manner as a native volume (such as local  
disk 618 (volume A)).

**Figure 7** shows an exemplary computer system acting as a server in accordance with the present invention. A server computer system 700 may be a desktop computer or any general or special purpose information processing device. The server system 5 700 features a system bus 713 for allowing components, such as a processor 712 and memory 711, to communicate with each other. Memory 711 consists of random access memory (RAM) or other volatile storage mechanism and operates to store instructions to be executed by processor 712 and data. Memory 711 also may be 10 used for storing temporary variables or other intermediate result data during execution of instructions by processor 712.

Server system 700 also has a bridge 714 which couples to an I/O (Input/Output) bus 715. I/O bus 715 connects to system 700 various peripheral and I/O devices such as a network interface 15 card (NIC) 716, a disk 718 and a CD-ROM (Compact Disc-Read Only Memory) 717. I/O bus 715 also couples to the system a display device 720 such as a monitor or LCD (Liquid Crystal Display). The NIC 716 enables the server 700 to connect to one or more networks 790 such as the Internet or a LAN (Local Area Network) 20 and facilitates the exchange of information between server 700 and other clients/servers on that network 790. Disk 718 is a hard disk drive, floppy disk drive or tape drive and is used to store data/application files and other more permanent information.

25 In accordance with the invention, disk 718 would house a profile datastore 740 (containing volume mounting parameter records for login IDs and for generic events) as well as software

to implement a web server. Using web server software, which can loaded into memory 711 and executed by processor 712, server 700 enables web pages and their components to be generated. For instance, when a client connects to the server website, the web

5 server would generate a dialog requesting the client for a login ID and password. The web server would then authenticate the supplied login ID and password and return corresponding volume mounting parameters to the plug-in on the client. If the profile datastore 740 is a database or a part thereof, database server

10 software may also be implemented in server 700 to process authentication and volume mounting parameter/profile retrievals. Server 700 is responsible for assembling the mounting command block which includes mounting parameters and a mount command. The server 700 also generates XML or other documents that may

15 contain one or more mounting command blocks and transmit these to clients via the client plug-in. The server 700 also has a network address which is obtained by the plug-in on the client. If the address of server 700 is on the approved list contained within the plug-in, then the volumes designated in the volume

20 mounting parameters of the mounting command block are mounted. Server 700 may be connected to one or more other servers such as a separate database server, or may incorporate a database server application as a process running within server 700.